



# Minería de datos y Patrones

Version 2024-I

## KNN

[ Capítulo 4 ]

**Dr. José Ramón Iglesias**

DSP-ASIC BUILDER GROUP

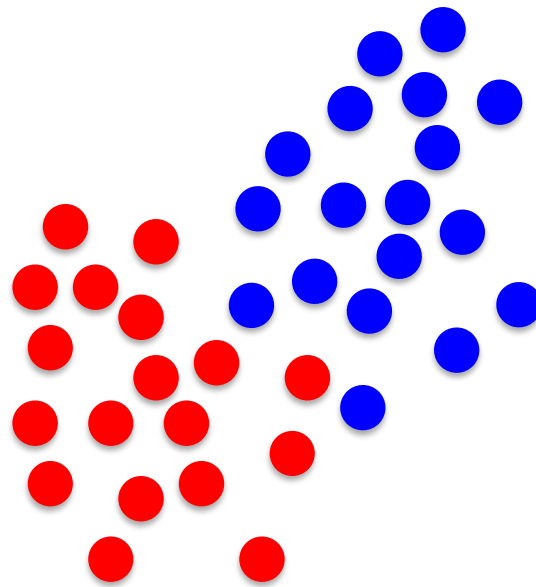
Director Semillero TRIAC

Ingeniería Electronica

Universidad Popular del Cesar

KNN: k nearest neighbors

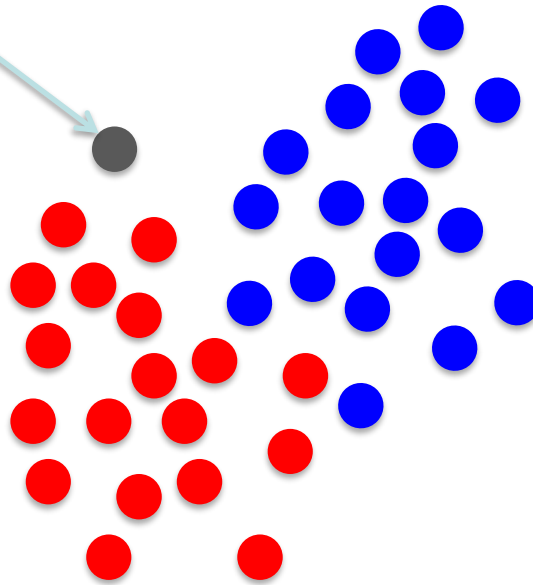
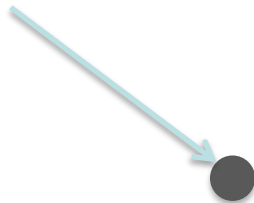
KNN: k nearest neighbors (two classes)



} Training data

KNN: k nearest neighbors (two classes)

Testing data  
?

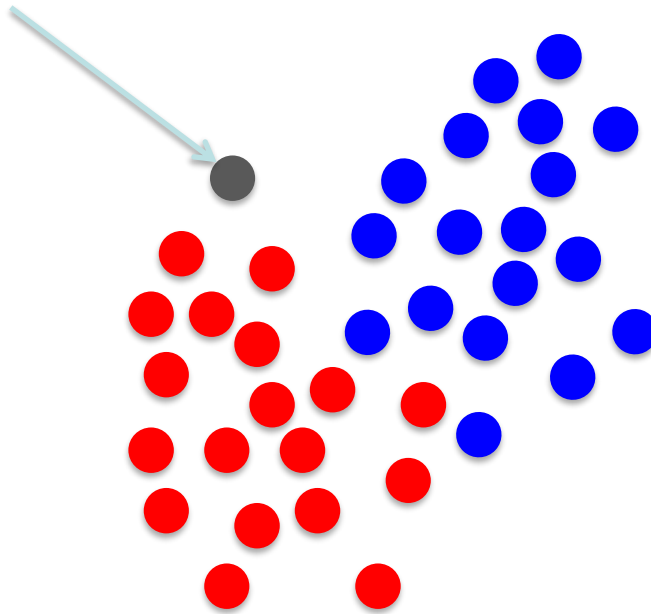


Training data

KNN: k nearest neighbors (two classes)

KNN Algorithm

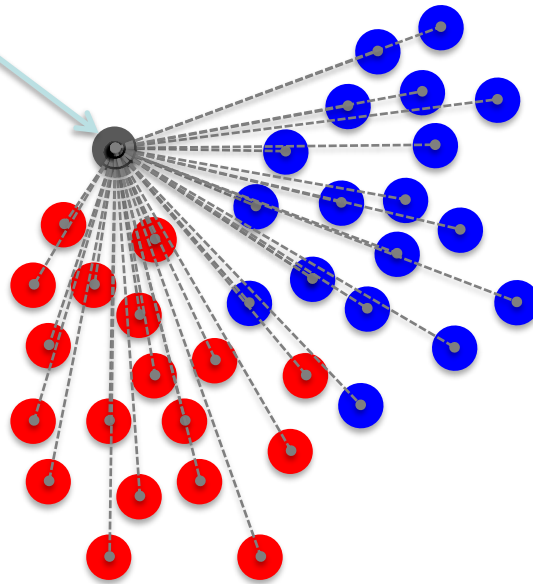
Testing data



# KNN: k nearest neighbors (two classes)

## KNN Algorithm

Testing data

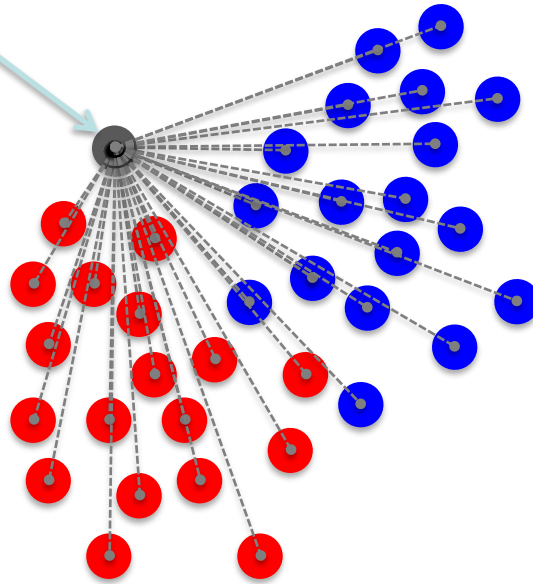


### 1. Distances

0.3655  
0.7015  
0.6712  
0.7474  
0.4313  
0.2880  
0.0115  
0.9202  
0.5304  
0.9362  
0.5989  
0.9447  
0.0569  
0.3264  
0.6811  
0.1332  
0.0226  
0.2435  
0.0705  
0.4839  
0.3631  
0.1090  
0.6296  
0.0508  
0.7660  
0.9544  
0.6487  
0.1519  
0.7936  
0.9525

# KNN: k nearest neighbors (two classes)

Testing data



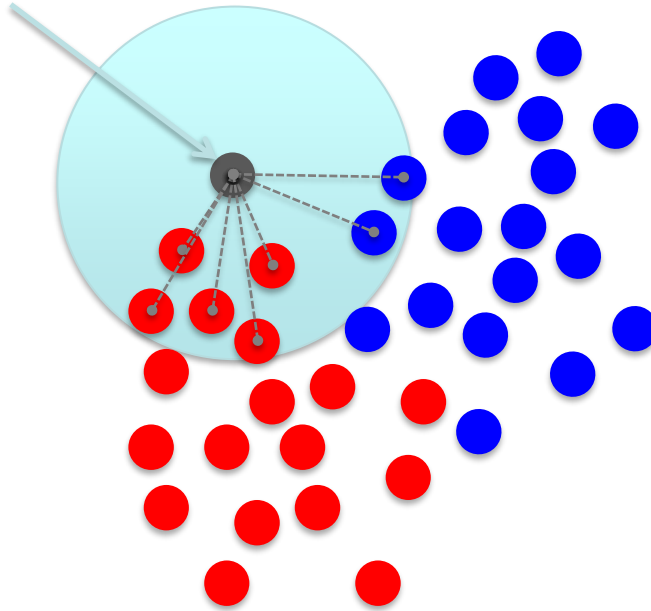
## KNN Algorithm

1. Distances
2. Sort

0.3655	0.0115
0.7015	0.0226
0.6712	0.0508
0.7474	0.0569
0.4313	0.0705
0.2880	0.1090
0.0115	0.1332
0.9202	0.1519
0.5304	0.2435
0.9362	0.2880
0.5989	0.3264
0.9447	0.3631
0.0569	0.3655
0.3264	0.4313
0.6811	0.4839
0.1332	0.5304
0.0226	0.5989
0.2435	0.6296
0.0705	0.6487
0.4839	0.6712
0.3631	0.6811
0.1090	0.7015
0.6296	0.7474
0.0508	0.7660
0.7660	0.7936
0.9544	0.9202
0.6487	0.9362
0.1519	0.9447
0.7936	0.9525
0.9525	0.9544

# KNN: k nearest neighbors (two classes)

Testing data



## KNN Algorithm

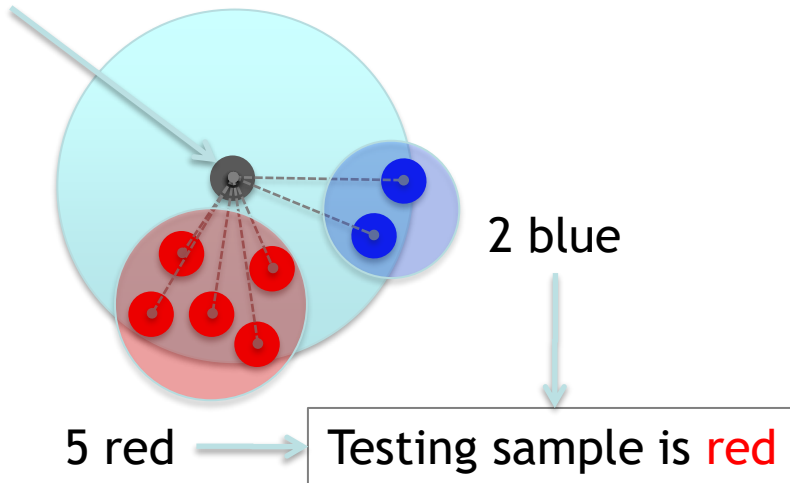
1. Distances
2. Sort
3. Take the k nearest  
(example k=7)

0.0115  
0.0226  
0.0508  
0.0569  
0.0705  
0.1090  
0.1332  
0.1519  
0.2435  
0.2880  
0.3264  
0.3631  
0.3655  
0.4313  
0.4839  
0.5304  
0.5989  
0.6296  
0.6487  
0.6712  
0.6811  
0.7015  
0.7474  
0.7660  
0.7936  
0.9202  
0.9362  
0.9447  
0.9525  
0.9544



# KNN: k nearest neighbors (two classes)

Testing data



## KNN Algorithm

1. Distances
2. Sort
3. Take the k nearest (example k=7)
4. Majority vote

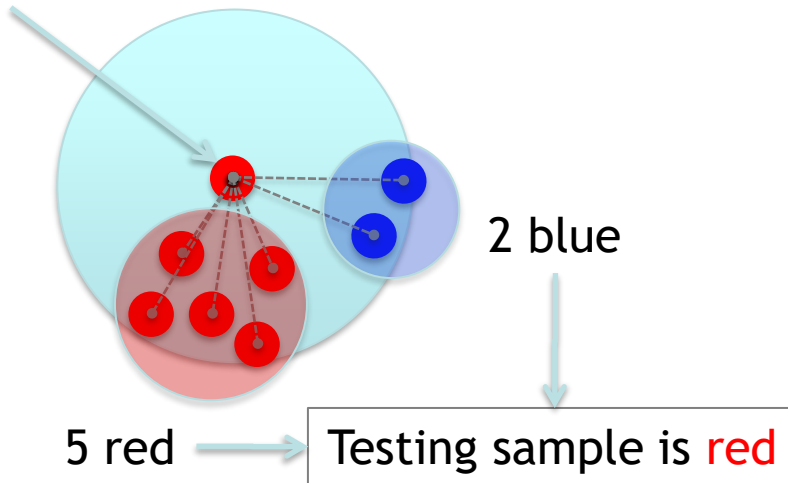
0.0115  
0.0226  
0.0508  
0.0569  
0.0705  
0.1090  
0.1332  
0.1519  
0.2435  
0.2880  
0.3264  
0.3631  
0.3655  
0.4313  
0.4839  
0.5304  
0.5989  
0.6296  
0.6487  
0.6712  
0.6811  
0.7015  
0.7474  
0.7660  
0.7936  
0.9202  
0.9362  
0.9447  
0.9525  
0.9544

# KNN: k nearest neighbors (two classes)

## KNN Algorithm

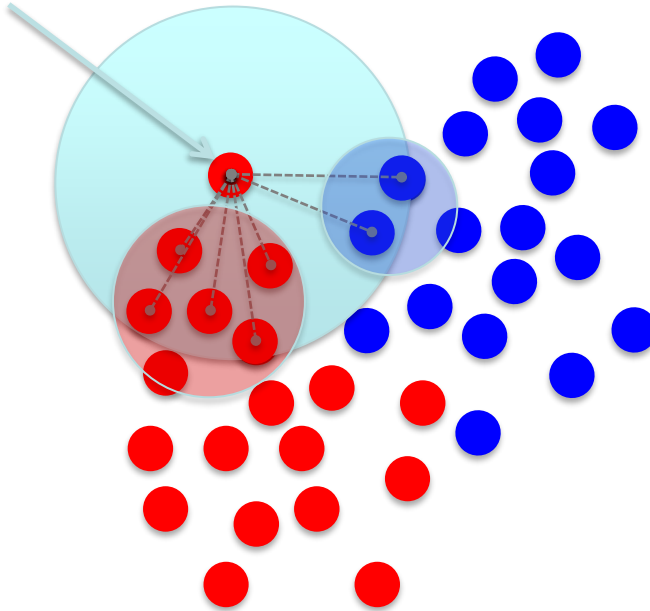
1. Distances
2. Sort
3. Take the k nearest (example k=7)
4. Majority vote

Testing data



## KNN: k nearest neighbors (two classes)

Testing data



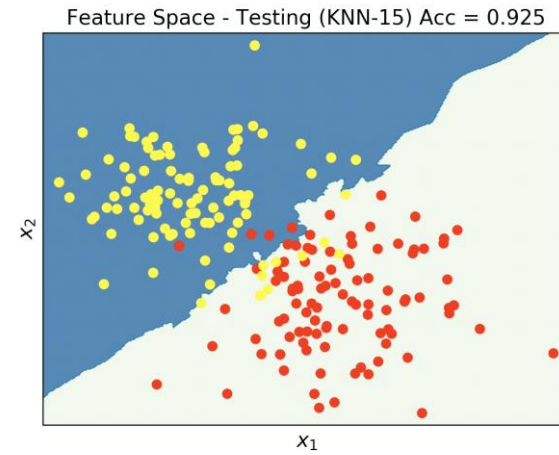
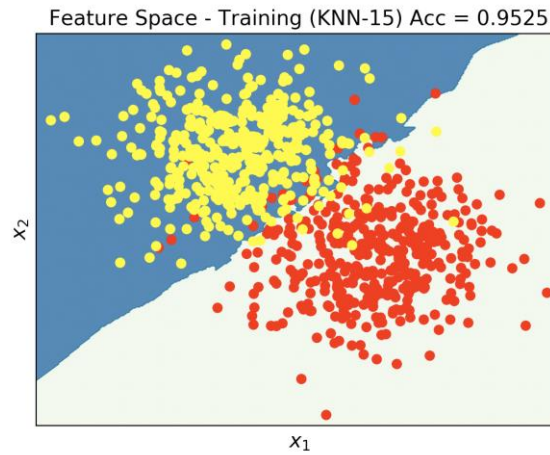
### KNN Algorithm

1. Distances
2. Sort
3. Take the k nearest  
(example  $k=7$ )
4. Majority vote

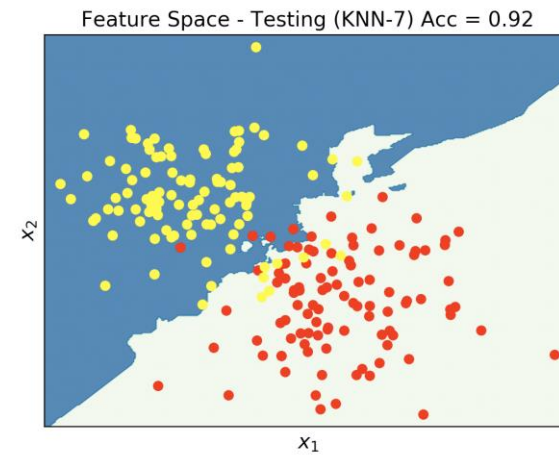
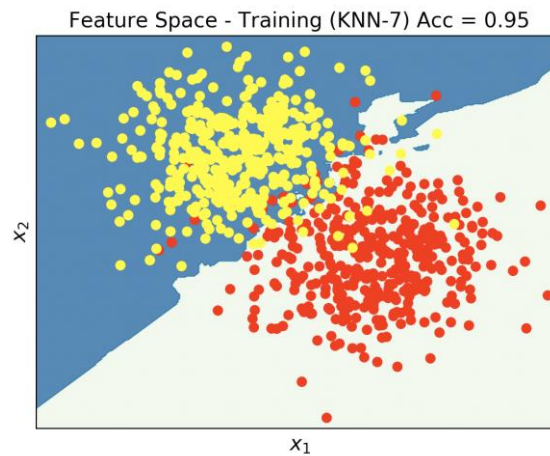
## TRAINING

## TESTING

$k = 15$



$k = 7$

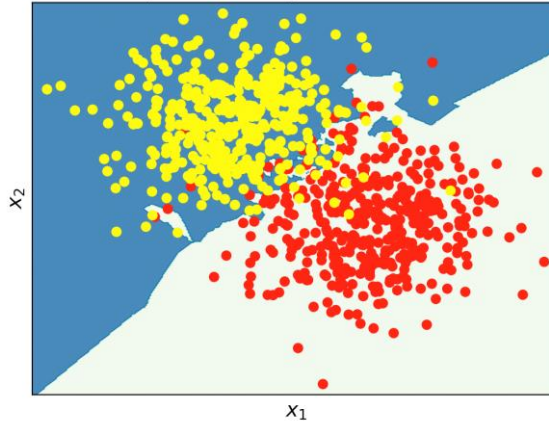


## TRAINING

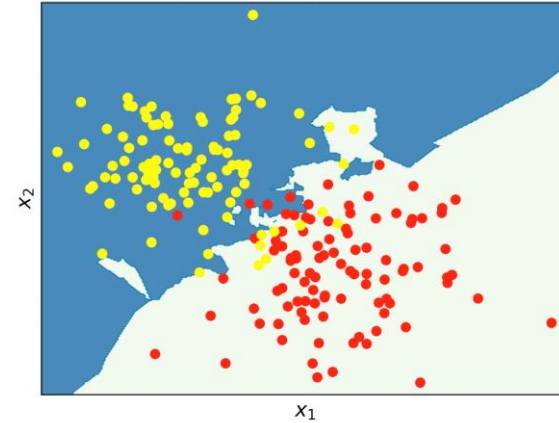
## TESTING

$k = 3$

Feature Space - Training (KNN-3) Acc = 0.95875

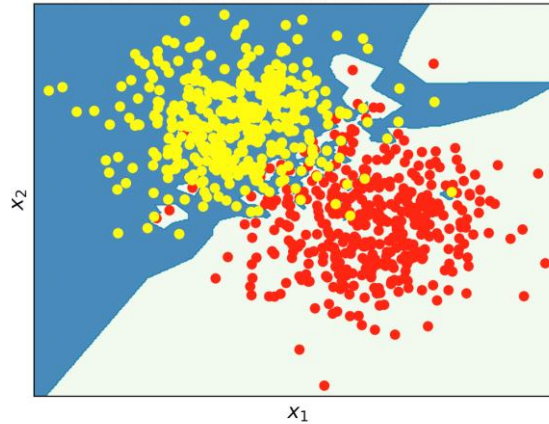


Feature Space - Testing (KNN-3) Acc = 0.915



$k = 1$

Feature Space - Training (KNN-1) Acc = 1.0



Feature Space - Testing (KNN-1) Acc = 0.905

